ReachFewL = ReachUL

Brady Garvin   Derrick Stolee*   Raghunath Tewari
N. V. Vinodchandran

August 15, 2011

The “Big” Theorems in Space-Bounded Complexity

The “Big” Theorems in Space-Bounded Complexity

- $\mathsf{NL} \subseteq \mathsf{SPACE}[\log^2(n)]$ Savitch, 1970

The "Big" Theorems in Space-Bounded Complexity

- $\mathsf{NL} \subseteq \mathsf{SPACE}[\log^2(n)]$ — Savitch, 1970
- $\mathsf{NL} = \mathsf{coNL}$ — Immerman-Szelepcsényi, 1987

The "Big" Theorems in Space-Bounded Complexity

- $\mathsf{NL} \subseteq \mathsf{SPACE}[\log^2(n)]$ — Savitch, 1970
- $\mathsf{NL} = \mathsf{coNL}$ — Immerman-Szelepcsényi, 1987
- $\mathsf{RL} \subseteq \mathsf{SC}^2$ — Nisan, 1995

The "Big" Theorems in Space-Bounded Complexity

- $\mathsf{NL} \subseteq \mathsf{SPACE}[\log^2(n)]$ Savitch, 1970
- $\mathsf{NL} = \mathsf{coNL}$ Immerman-Szelepcsényi, 1987
- $\mathsf{RL} \subseteq \mathsf{SC}^2$ Nisan, 1995
- $\mathsf{RL} \subseteq \mathsf{SPACE}[\log^{3/2}(n)]$ Saks, Zhou, 1999

The “Big” Theorems in Space-Bounded Complexity

- $\mathsf{NL} \subseteq \mathsf{SPACE}[\log^2(n)]$ — Savitch, 1970
- $\mathsf{NL} = \mathsf{coNL}$ — Immerman-Szelepcsényi, 1987
- $\mathsf{RL} \subseteq \mathsf{SC}^2$ — Nisan, 1995
- $\mathsf{RL} \subseteq \mathsf{SPACE}[\log^{3/2}(n)]$ — Saks, Zhou, 1999
- $\mathsf{L} = \mathsf{SL}$ — Reingold, 2008

The "Big" Theorems in Space-Bounded Complexity

- $\mathsf{NL} \subseteq \mathsf{SPACE}[\log^2(n)]$ — Savitch, 1970
- $\mathsf{NL} = \mathsf{coNL}$ — Immerman-Szelepcsényi, 1987
- $\mathsf{RL} \subseteq \mathsf{SC}^2$ — Nisan, 1995
- $\mathsf{RL} \subseteq \mathsf{SPACE}[\log^{3/2}(n)]$ — Saks, Zhou, 1999
- $\mathsf{L} = \mathsf{SL}$ — Reingold, 2008
- $\mathsf{NL} \stackrel{?}{=} \mathsf{UL}$ — ???, 20??

Unambiguous Log-Space

UL contains languages A with non-deterministic log-space machines so that

Unambiguous Log-Space

UL contains languages A with non-deterministic log-space machines so that

- For each $x \notin A$, there are no accepting computation paths.

Unambiguous Log-Space

UL contains languages A with non-deterministic log-space machines so that

- For each $x \notin A$, there are no accepting computation paths.
- For each $x \in A$ there is exactly one accepting computation path from the initial configuration.

Log-space Classes and Reachability

L
Deterministic

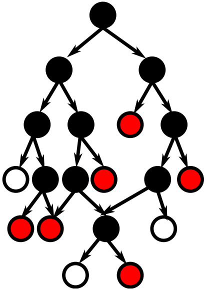
Complete:
Undirected Reach
(Reingold 08)

Log-space Classes and Reachability

L
Deterministic

Complete:
Undirected Reach
(Reingold 08)

NL
Nondeterministic



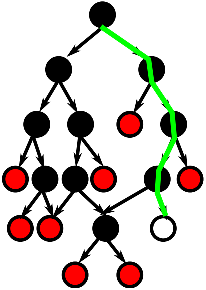
Complete:
Directed Reach

Log-space Classes and Reachability

L
Deterministic

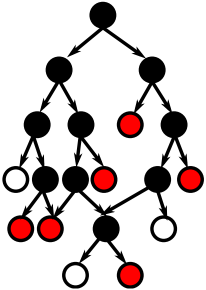
Complete:
Undirected Reach
(Reingold 08)

UL
Unambiguous



Contains:
Dir. Planar Reach
(BTV 09)

NL
Nondeterministic



Complete:
Directed Reach

Much ado about UL

Much ado about UL

- $\mathsf{UL}/\mathrm{poly} = \mathsf{NL}/\mathrm{poly}$. Reinhardt, Allender, 2002

Much ado about UL

- UL/poly $=$ NL/poly. Reinhardt, Allender, 2002

- Planar Reachability is in UL Bourke, T—, V—, 2009

FewL

FewL contains languages A with non-deterministic Turing machines and a constant k so that

FewL

FewL contains languages A with non-deterministic Turing machines and a constant k so that

- For $x \notin A$, there are no accepting computation paths.

FewL

FewL contains languages A with non-deterministic Turing machines and a constant k so that

- For $x \notin A$, there are no accepting computation paths.
- For $x \in A$, there are at most n^k accepting computation paths.

FewL

FewL contains languages A with non-deterministic Turing machines and a constant k so that

- For $x \notin A$, there are no accepting computation paths.
- For $x \in A$, there are at most n^k accepting computation paths.

$$\mathrm{UL} \subseteq \mathrm{FewL} \subseteq \mathrm{NL}$$

Other Forms of Unambiguity

- UL has *at most one* path from **initial** configuration to **accepting** configuration.

Other Forms of Unambiguity

- UL has *at most one* path from **initial** configuration to **accepting** configuration.
- ReachUL has *at most one* path from **initial** configuration to **any** configuration.

Other Forms of Unambiguity

- UL has *at most one* path from **initial** configuration to **accepting** configuration.
- ReachUL has *at most one* path from **initial** configuration to **any** configuration.
- StrongUL has *at most one* path from **any** configuration to **any** configuration.

Other Forms of Unambiguity

- UL has *at most one* path from **initial** configuration to **accepting** configuration.
- ReachUL has *at most one* path from **initial** configuration to **any** configuration.
- StrongUL has *at most one* path from **any** configuration to **any** configuration.

$$\mathsf{StrongUL} \subseteq \mathsf{ReachUL} \subseteq \mathsf{UL}$$

Other Forms of Fewness

- FewL has *polynomially many* paths from **initial** configuration to **accepting** configuration.

Other Forms of Fewness

- FewL has *polynomially many* paths from **initial** configuration to **accepting** configuration.
- ReachFewL has *polynomially many* paths from **initial** configuration to **any** configuration.

Other Forms of Fewness

- FewL has *polynomially many* paths from **initial** configuration to **accepting** configuration.
- ReachFewL has *polynomially many* paths from **initial** configuration to **any** configuration.
- StrongFewL has *polynomially many* paths from **any** configuration to **any** configuration.

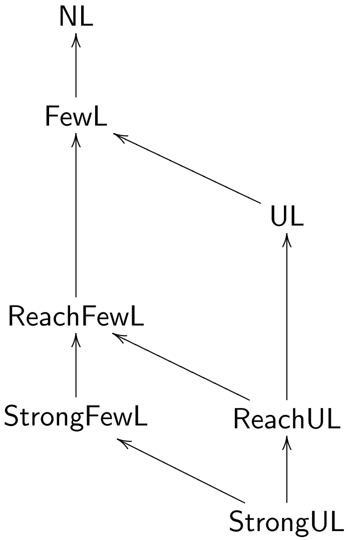
Other Forms of Fewness

- FewL has *polynomially many* paths from **initial** configuration to **accepting** configuration.
- ReachFewL has *polynomially many* paths from **initial** configuration to **any** configuration.
- StrongFewL has *polynomially many* paths from **any** configuration to **any** configuration.

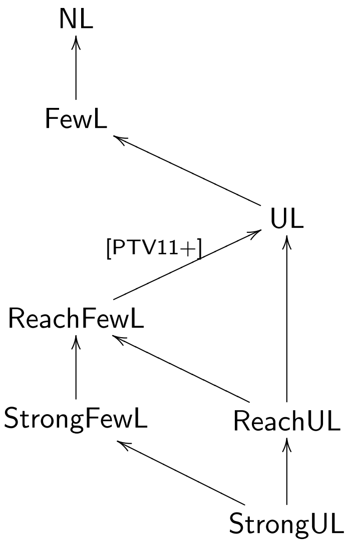
$$\text{StrongFewL} \subseteq \text{ReachFewL} \subseteq \text{FewL}$$

(ReachUL and ReachFewL originally defined by Buntrock, Jenner, Lange, and Rossmanith in 1991.)

Unambiguous Complexity Classes



Unambiguous Complexity Classes



Previous Results with ReachUL and ReachFewL

- ReachUL is closed under complement. BJLR, 1991

Previous Results with ReachUL and ReachFewL

- ReachUL is closed under complement. BJLR, 1991
- ReachUL has a complete problem. Lange, 1997

Previous Results with ReachUL and ReachFewL

- ReachUL is closed under complement. BJLR, 1991
- ReachUL has a complete problem. Lange, 1997
- ReachUL $\subseteq$ SPACE $\left[\frac{\log^2(n)}{\log\log(n)}\right]$. Allender, Lange, 1998

Previous Results with ReachUL and ReachFewL

- ReachUL is closed under complement. BJLR, 1991
- ReachUL has a complete problem. Lange, 1997
- ReachUL $\subseteq \mathrm{SPACE}\left[\frac{\log^2(n)}{\log\log(n)}\right]$. Allender, Lange, 1998
- ReachFewL $\subseteq$ UL $\cap$ coUL Pavan, T—, V—, 2011

Complete Problems

$$L_{ru} = \{\langle G, s, t\rangle : G \text{ is a graph with exactly one path from } s \text{ to } t,$$
$$\text{there is at most one path from } s \text{ to any other vertex in } G\}.$$

L_{ru} is ReachUL-complete.

Complete Problems

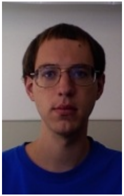
$$L_{ru} = \{\langle G, s, t\rangle : G \text{ is a graph with exactly one path from } s \text{ to } t,$$
$$\text{there is at most one path from } s \text{ to any other vertex in } G\}.$$

L_{ru} is ReachUL-complete.

$$L_{rf}^{(k)} = \{\langle G, s, t\rangle : G \text{ is a graph with a path from } s \text{ to } t,$$
$$\text{there are at most } n^k \text{ paths from } s \text{ to any other vertex in } G\}.$$

Each language in ReachFewL reduces to $L_{rf}^{(k)}$ for some k.

Main Theorem



Brady Garvin

Derrick Stolee

Raghunath Tewari

N. V. Vinodchandran

$$\mathsf{ReachFewL} = \mathsf{ReachUL}$$

A Lemma About Oracles

Lemma

$\mathsf{L}^{\mathsf{ReachUL}} = \mathsf{ReachUL}$

Proof.

We can assume O is a ReachUL-complete oracle.

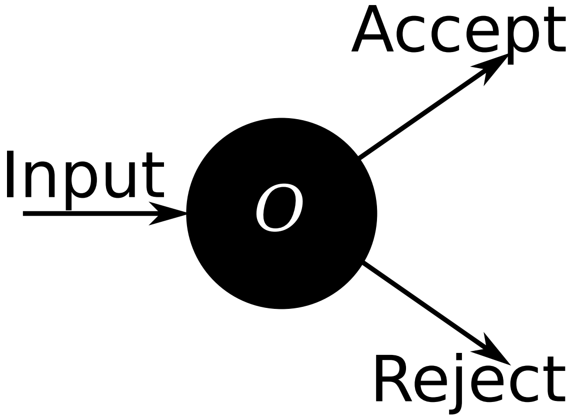
There are two terminal configurations: "accept" and "reject."

A Lemma About Oracles

Lemma

$\mathsf{L}^{\mathsf{ReachUL}} = \mathsf{ReachUL}$

Proof.

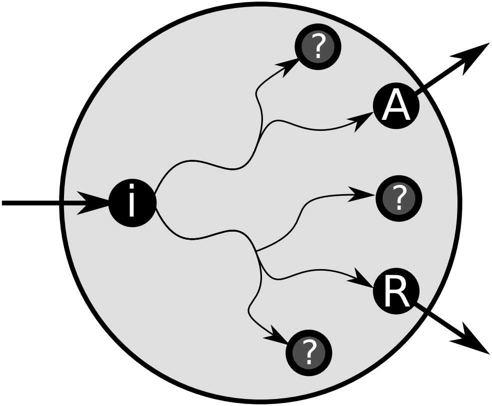


A Lemma About Oracles

Lemma

$L^{\text{ReachUL}} = \text{ReachUL}$

Proof.

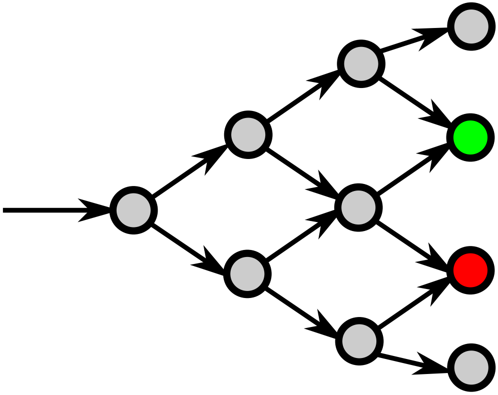


A Lemma About Oracles

Lemma

$\mathsf{L}^{\mathsf{ReachUL}} = \mathsf{ReachUL}$

Proof.

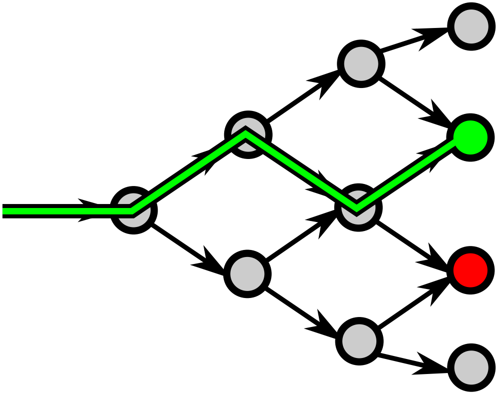


A Lemma About Oracles

Lemma

$\mathrm{L}^{\mathrm{ReachUL}} = \mathrm{ReachUL}$

Proof.



Distance Isolation

A (weighted) graph G with vertex s is *distance-isolated* with respect to s if there is no vertex t so that there are two paths from s to t of the same weight.

A Hashing Result

Theorem (Fredman, Komlós and Szemerédi, 1984)

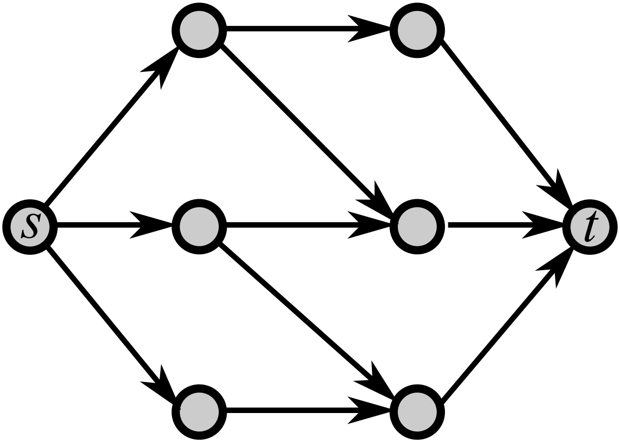
Let c be a constant and S be a set of n-bit integers with $|S| \leq n^c$. Then there is a c' and a $(c' \log n)$-bit prime number p so that for any $x \neq y \in S$, we have $x \not\equiv y \pmod{p}$.

Hashing to Distance Isolation

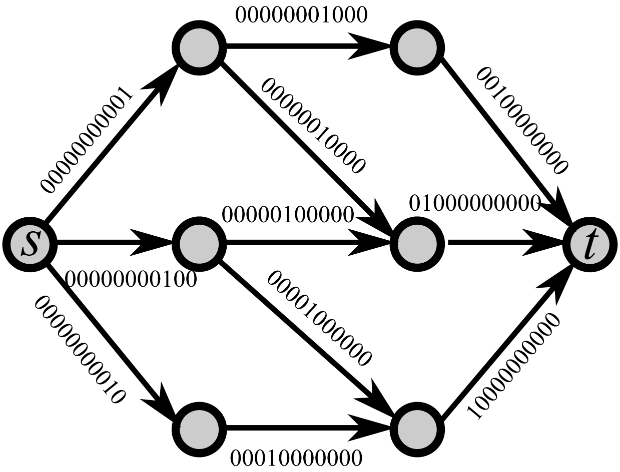
Lemma

Let G be a graph with edges $E(G) = \{e_1, e_2, \ldots, e_\ell\}$. If G has at most n^k paths from u to any vertex $v \in V(G)$, then there is a prime $p \leq n^{k'}$, for some constant k', such that the weight function $w_p : E(G) \to \{1, \ldots, p\}$ given by $w_p(e_i) = 2^i \pmod{p}$ defines a weighted graph G_{w_p} which is distance isolated with respect to u.

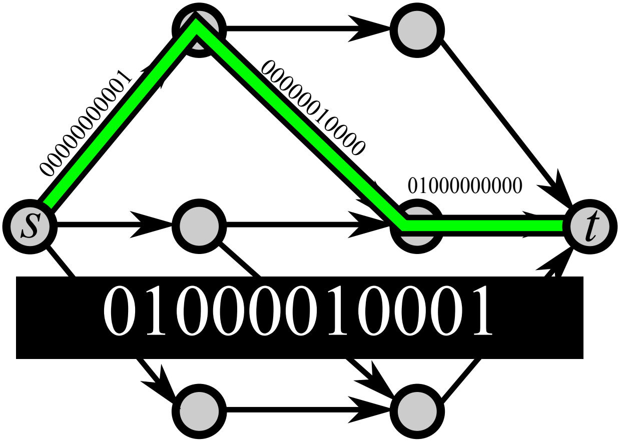
Hashing to Distance Isolation



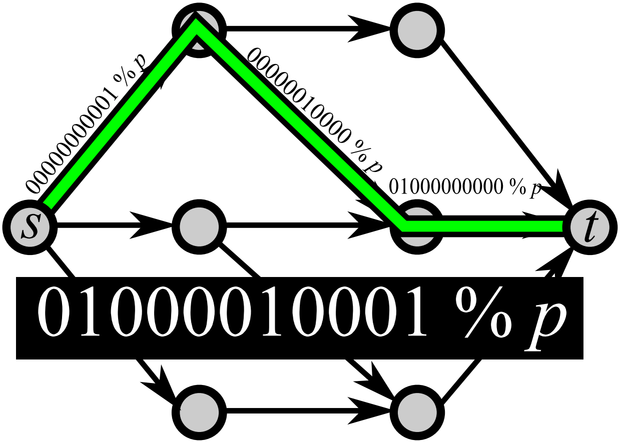
Hashing to Distance Isolation



Hashing to Distance Isolation



Hashing to Distance Isolation

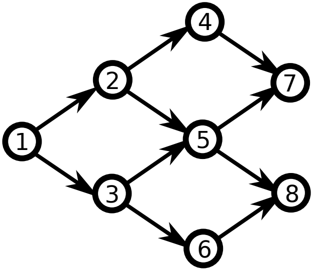


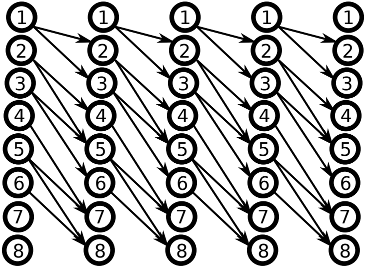
Layering Transformation

Let G be a graph. The *layered graph* $\mathrm{lay}(G)$ is the graph on vertex set $V(G) \times \{0, \ldots, n(G)\}$ with edges $(u, i) \longrightarrow (v, i + 1)$ whenever $u \longrightarrow v$ is in G.

Layering Transformation

Let G be a graph. The *layered graph* $\mathrm{lay}(G)$ is the graph on vertex set $V(G) \times \{0, \dots, n(G)\}$ with edges $(u, i) \longrightarrow (v, i + 1)$ whenever $u \longrightarrow v$ is in G.





Layering Transformation

Lemma

There is a path of distance d from s to t in G if and only if there is a path from $(s, 0)$ to (t, d) in $\mathrm{lay}(G)$.

Layering Transformation

Lemma

There is a path of distance d from s to t in G if and only if there is a path from $(s, 0)$ to (t, d) in $\mathrm{lay}(G)$.

Corollary

G is distance-isolated with respect to s if and only if $\mathrm{lay}(G)$ is reach-unique with respect to $(s, 0)$.

Putting it Together

1. Given a reach-n^k graph G with vertices s, t.

Putting it Together

1. Given a reach-n^k graph G with vertices s, t.
2. For each prime $p \in \{2, 3, \ldots, n^{k'}\}$, generate G_{w_p}.

Putting it Together

1. Given a reach-n^k graph G with vertices s, t.
2. For each prime $p \in \{2, 3, \ldots, n^{k'}\}$, generate G_{w_p}.
3. Generate lay(G_{w_p}).

Putting it Together

1. Given a reach-n^k graph G with vertices s, t.
2. For each prime $p \in \{2, 3, \ldots, n^{k'}\}$, generate G_{w_p}.
3. Generate $\mathrm{lay}(G_{w_p})$.
4. Test (using ReachUL) if $\mathrm{lay}(G_{w_p})$ is reach-unique.

Putting it Together

1. Given a reach-n^k graph G with vertices s, t.
2. For each prime $p \in \{2, 3, \ldots, n^{k'}\}$, generate G_{w_p}.
3. Generate $\mathrm{lay}(G_{w_p})$.
4. Test (using ReachUL) if $\mathrm{lay}(G_{w_p})$ is reach-unique.
5. If so, test if $(s, 0) \rightarrow (t, d)$ exists for each distance d.

Putting it Together

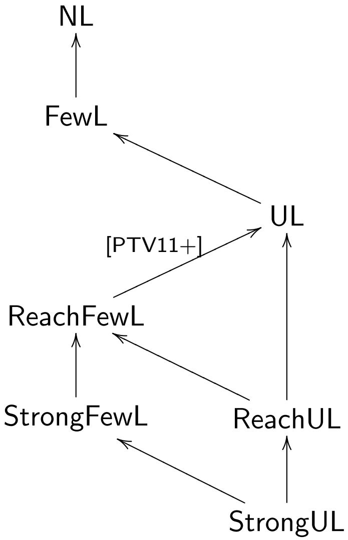
1. Given a reach-n^k graph G with vertices s, t.
2. For each prime $p \in \{2, 3, \ldots, n^{k'}\}$, generate G_{w_p}.
3. Generate $\mathrm{lay}(G_{w_p})$.
4. Test (using ReachUL) if $\mathrm{lay}(G_{w_p})$ is reach-unique.
5. If so, test if $(s, 0) \to (t, d)$ exists for each distance d.
6. If all attempts fail, reject.

ReachFewSearch(G, s, t)

Require: G has at most n^k paths from s to any other vertex.
Ensure: Accepts if and only if there is a path from s to t in G.

```
for all primes p ∈ {1, ..., n^{k'}} do
    Define w_p(e_i) = 2^i (mod p).
    Construct G_{w_p}.
    Construct lay(G_{w_p}).
    if IsReachUnique(lay(G_{w_p})) then
        for each d ∈ {1, ..., n(G_{w_p})} do
            if ReachUnique(lay(G_{w_p}), (s, 0), (t, d)) then
                return True
            end if
        end for
        return False
    end if
end for
return False
```

Unambiguous Complexity Classes: Before



Unambiguous Complexity Classes: After

$$\begin{array}{c}
\mathsf{NL} \\
\uparrow \\
\mathsf{FewL} \\
\uparrow \\
\mathsf{UL} \\
\uparrow \\
\mathsf{ReachFewL} = \mathsf{ReachUL} \\
\uparrow \\
\mathsf{StrongFewL} \\
\uparrow \\
\mathsf{StrongUL}
\end{array}$$

ReachFewL = ReachUL

Brady Garvin   Derrick Stolee*   Raghunath Tewari
N. V. Vinodchandran

August 15, 2011

This work is supported in part by NSF Grants CCF-0916525, CFDA#47.076, DMS-0914815, and AFOSR grant FA9550-10-1-0406.